BAB II
LANDASAN TEORI

2.1. Sistem Informasi

2.1.1. Pengertian Sistem Informasi

Terdapat beberapa definisi yang dipaparkan oleh para ahli mengenai sistem informasi, yaitu:

1. Menurut Alter (1992), sistem informasi adalah kombinasi antara prosedur kerja, informasi, orang, dan teknologi informasi yang diorganisasikan untuk mencapai tujuan dalam sebuah organisasi.
2. Menurut Bodnar dan Hopwood (1993), sistem informasi adalah kumpulan perangkat keras dan perangkat lunak yang dirancang untuk mentransformasikan data kedalam bentuk informasi yang berguna.
3. Menurut Hall (2001), sistem informasi adalah sebuah rangkaian prosedur formal dimana data dikelompokkan, diproses menjadi informasi, dan didistribusikan kepada pemakai.

Dari berbagai definisi diatas, dapat disimpullkan bahwa sistem informasi merupakan kumpulan komponen seperti manusia, komputer, teknologi informasi, dan prosedur kerja yang saling berhubungan antara satu dengan yang lainnya yang membentuk satu kesatuan untuk melakukan proses pengolahan data menjadi informasiuntuk mencapai tujuan tertentu. (Abdul Kadir, 2003).

2.1.2. Jenis-jenis Sistem Informasi

Sistem informasi dibangun dengan tujuan dan fungsi yang berbeda-beda. Untuk itu, sistem informasi tediri atas beberapa jenis, yaitu: (Sutejo dan Budi, 2002).

1. *Transaction Processing Sistems* (TPS)

TPS adalah sistem informasi terkomputerisasi yang dibangun untuk transaksi bisnis rutin seperti daftar gaji dan inventarisasi. TPS berfungsi pada

level organisasi yang memungkinkan organisasi bisa berinteraksi dengan lingkungan eksternal. Data atau informasi yang dihasilkan oleh TPS dapat dilihat atau digunakan oleh manajer.

2. *Office Automation Sistems* (OAS) dan *Knowledge Work Sistems* (KWS)

OAS mendukung pekerja data, yang biasanya tidak menciptakan pengetahuan baru, tetapi hanya menganalisa informasi yang nantinya akan disebarkan secara keseluruhan dengan organisasi dan kadang-kadang diluar organisasi. Aspek-aspek OAS seperti word processing, spreadsheets, electronic scheduling, dan komunikasi melalui voice mail, email dan video conferencing.

KWS mendukung para pekerja profesional seperti ilmuwan, insinyur dan doktor dengan membantu menciptakan pengetahuan baru dan memungkinkan mereka mengkontribusikannya ke organisasi atau masyarakat.

3. Sistem Informasi Manajemen (SIM)

SIM mendukung spektrum tugas-tugas organisasional yang lebih luas dari TPS, termasuk menganalisa keputusan dan membuat keputusan. SIM menghasilkan informasi yang digunakan untuk membuat keputusan, dan juga dapat membatu menyatukan beberapa fungsi informasi bisnis yang sudah terkomputerisasi.

4. *Decision Support Sistems* (DSS)

DSS hampir sama dengan SIM karena menggunakan basis data sebagai sumber data yang di olah. DSS berawal dari SIM yang menekankan pada fungsi mendukung pengambilan keputusan diseluruh tahap-tahapnya, meskipun pada akhirnya keputusan aktual tetap wewenang eksklusif pembuat keputusan.

5. *Expert Sistems* (ES) dan *Artificial Intelligence* (AI)

AI dibangun untuk mengembangkan mesin-mesin yang mengadopsi kecerdasan manusia. Dalam melakukan riset AI, hal-hal yang perlu dilakukan adalah memahami bahasa alamiahnya dan menganalisa kemampuannya untuk berfikir melalui problem sampai kesimpulan logiknya. Yang mana semua itu ditujukan untuk menyelesaikan suatu permasalahan.

Sistem ahli disebut juga sebagai *knowledge-based sistems*, karena secara efektif menangkap dan menggunakan pengetahuan seorang ahli untuk menyelesaikan masalah yang dialami dalam suatu organisasi. Berbeda dengan DSS yang meningalkan keputusan terakhir bagi pembuat keputusan, sistem ahli menyeleksi solusi terbaik terhadap suatu masalah khusus. Komponen dasar sistem ahli adalah *knowledge-base* yakni suatu mesin inferensi yang menghubungkan pengguna dengan sistem melalui pengolahan pertanyaan lewat bahasa terstruktur dan antarmuka pengguna.

6. *Group Decision Support Sistems* (GDSS)

GDSS dimaksudkan untuk membawa suatu kelompok menyelesaikan masalah secara bersama-sama dengan memberi bantuan dalam bentuk pendapat, kuesioner, konsultasi dan skenario. Terkadang GDSS disebut juga dengan CSCW yang mencakup pendukung perangkat lunak yang disebut dengan "*groupware*" untuk kolaborasi tim melalui komputer yang terhubung dengan jaringan.

7. *Executive Support Sistems* (ESS)

ESS dibangun untuk membantu eksekutif dalam mengatur interaksinya dengan lingkungan eksternal dengan menyediakan grafik-grafik dan pendukung komunikasi di tempat-tempat yang bisa diakses seperti kantor.

2.1.3. Pengembangan Sistem Informasi

Dalam membangun sistem informasi, seorang *IT Engineer* tidak hanya mengotomatisasi prosedur lama, tetapi juga menata dan memperbaharui juga menciptakan aliran data yang baru yang lebih efisien. Untuk mengembangkan sistem informasi dibutuhkan suatu metodologi yang disebut dengan metodologi pengembangan sistem. Metodologi tersebut merupakan suatu proses standar yang diikuti oleh organisasi untuk melaksanakan seluruh langkah yang diperlukan untuk menganalisa, merancang, mengimplementasikan, dan memelihara sistem informasi. (Hoffer dkk, 1998)

Terdapat beberapa metode yang umum digunakan dalam pengembangan sistem informasi, yaitu *System Development Life Cycle* dan *Prototipe.* (Abdul kadir, 2003).

Metode SDLC(*System Development Life Cycle*) atau yang dikenal dengan *waterfall* merupakan metode klasik yang digunakan untuk mengembangkan, memelihara, dan menggunakan sistem informasi. Tahapan pada model *waterfall* terdiri dari: (Sutejo dan Budi, 2002).

1. Analisis Sistem

Pada tahap ini akan dilakukan proses menganalisis dan mendefinisikan masalah dan kemungkinan solusinya untuk sistem informasi dan proses organisasi. Tujuan dari tahap ini adalah untuk menentukan hal-hal detail tentang yang akan dikerjakan oleh sistem yang diusulkan. Proses ini mencakup studi kelayakan dan analisa kebutuhan.

Studi kelayakan digunakan untuk menentukan kemungkinan keberhasilan solusi yang diusulkan untuk memastikan bahwa solusi yang diusulkan dapat benar-benar tercapai. Terdapat beberapa aspek yang perlu diperhatikan dalam menentukan kelayakan suatu solusi, yaitu, teknologi, ekonomi, non ekonomi, organisasi, jadwal, serta kendala hukum dan etika. Sementara itu, analisa kebutuhan dilakukan untuk menghasilkan spesifikasi kebutuhan sistem yang akan dikembangkan. (Abdul Kadir, 2003).

2. Perancangan Sistem

Pada tahap ini pengembang sistem akan merancang output, input, struktur file, program, prosedur, perangkat keras dan perangkat lunak yang diperlukan untuk mendukung sistem informasi. Hasil akhirnya berupa spesifikasi rancangan yang sangat rinci untuk memudahkan *programmer* dalam pengembangan sistem informasi.

3. Pembangunan dan Testing Sistem

Pada tahap ini pemrogram akan melakukan pemrograman untuk membangun perangkat lunak yang diperlukan dalam mendukung sistem. Kemudian dilakukan testing secara akurat. Melakukan instalasi dan testing terhadap perangkat keras dan mengoperasikan perangkat lunak.

4. Implementasi Sistem

Pada tahap ini sistem yang sudah dibangun akan diterapkan. Pada tahap ini juga dilakukan pelatihan dan panduan yang diperlukan pengguna yang akan menggunakan sistem.

5. Operasi dan Perawatan

Pada tahap ini pengembang sistem akan melakukan perubahan atau tambahan fasilitas.

6. Evaluasi Sistem

Pengembang sistem akan mengevaluasi sejauh mana sistem telah dibangun dan seberapa bagus sistem telah dioperasikan.

Tahapan diatas merupakan model *waterfall* dalam pengembangan sistem informasi. Model-model baru yang sekarang ini marak digunakan, seperti prototyping, spiral, 4GT dan kombinasi dikembangkan dari model di atas.

2.2. Artificial Intelligence

Artificial Intelligence merupakan ilmu komputer yang mempelajari proses bagaimana komputer dapat melaksanakan kejadian-kejadian atau menyelesaikan suatu masalah dengan menggunakan pemikiran atau kecerdasan seperti layaknya manusia(Kadek Ayu Yanti Pawitri, Joko Purwadi 2007).

Teknik pemrograman dengan kecerdasan buatan melakukan prosesnya dengan menirukan apa yang dilakukan oleh otak manusia. Selan itu, kecerdasan buatan juga meniru proses belajar manusia bagaimana manusia menyerap informasi yang baru yang akan digunakan sebagai referensi pada waktu yang akan datang. Informasi tersebut dapat disimpan tanpa harus mengubah cara kerja pikiran atau mengganggu seluruh fakta-fakta yang sudah ada. Sehingga dengan kecerdasan buatan dimungkinkan untuk membuat program di mana setiap bagian dari program benar-benar independen.

Cakupan pembahasan *Artificial Intelligence* meliputi beberapa hal, yaitu *Logic, Searching, Vision, Recognition* dan *Pattern Matching, Natural Language*

Processing (NLP), Robotik, *Learning*, *Uncertainty* dan Fuzzy *Logic*.(Suyanto, 2007).

1. *Logic*

Program dapat digunakan untuk mempelajari perbaikan logika dari sebuah argumen dengan menerapkan aturan logika standar.

2. *Searching*

Diterapkan pada AI yang mengacu pada pencarian untuk menemukan solusi dalam penyelesaian sebuah masalah.

3. *Vision, Recognition* dan *Pattern Matching*

Penting untuk beberapa aplikasi, termasuk robotik dan pengolah citra(*image processing*). Pada hal ini dibutuhkan untuk memperbolehkan computer berhubungan secara langsung ke dunia dan manusia. Jika computer berhubungan secara menyeluruh dengan dunia manusia, maka dibutuhkan beberapa kemampuan bayangan (*vision*).

4. *Natural Language Processing* (NLP)

Bagian yang paling sulit dari sasaran AI untuk mendapatkannya karena NLP memperbolehkan komputer untuk mengerti bahasa manusia secara langsung.

5. Robotik

Digunakan untuk mempelajari mengontrol gerakan suatu perangkat yang telah dirancang.

6. *Learning*

Bertransaksi dengan pembuatan program yang belajar dari kesalahan dari observasi atau permintaan komputer mempunyai kemampuan untuk mengambil keuntungan dari pengalaman.

7. *Uncertainty* (Ketidak Pastian) dan *Fuzzy Logic*

Komputer dapat berpikir dengan menggunakan pengetahuan yang tidak lengkap dengan menerapkan penggunaan *Fuzzy Logic. Fuzzy logic* juga dapat menyelesaikan masalah yang mengandung keraguan, ketidaktepatan, dan kebenaran yang bersifat sebagian.

2.3. Searching

Searching merupakan salah satu teknik dalam menyelesaikan masalah melalui penelusuran kemungkinan-kemungkinan untuk mendapatkan suatu solusi. Langkah pertama yang harus dilakukan untuk menyelesaikan masalah dengan teknik *searching* adalah mendefinisikan ruang masalah. Ruang masalah dapat di gambarkan sebagai himpunan rute penelusuran dari keadaan awal sampai keadaan akhir (*goal state).* Yang mana ruang masalah yang telah dibentuk nantinya akan bertindak sebagai *track* yang akan ditelusuri hingga solusi ditemukan.

Langkah selanjutnya adalah mendefinisikan aturan produksi yang digunakan untuk mengubah *state* satu ke *state* lainnya. Dan langkah yang terakhir adalah memilih metode yang tepat untuk proses penelusuran. Secara umum, metode pencaria terdiri dari dua jenis, yaitu *blind search* dan *Heuristic Search.*(Suyanto, 2007).

2.3.1. Blind Search

Blind Search adalah pencarian yang melakukan proses penelusuran tanpa dibekali dengan informasi. Artinya pencarian akan meraba setiap state satu persatu hingga solusi ditemukan dan membangkitkan simpul berikutnya hanya berdasarkan urutan tertentu. Terdapat beberapa jenis algoritma pencarian yang tergolong jenis ini, yaitu *Breath First Search(BFS), Uniform Cost Search(UCS), Depth First Search(DFS), Dept Limited Search(DLS),* dan *Iterative Deepening Search(IDS).*

2.3.2. Heuristic Search

Heuristic Search adalah pencarian yang melakukan proses penelusuran dibekali dengan informasi. Artinya pencarian akan membangkitkan state berdasarkan informasi yang berupa jarak heuristik yaitu biaya perkiraan dari simpul tertentu ke simpul tujuan. Pada jenis ini terdapat fungsi yang jarak heuristik. Fungsi itu adalah fungsi heuristik. Metode perhitungan jarak heuristik yang umum digunakan terdiri dari dua jenis, yaitu Manhattan Distance dan Euclidean Distance.

1. Manhattan Distance

Manhattan Distance merupakan fungsi heuristik yang paling umum digunakan. Fungsi ini hanya akan menjumlahkan selisih nilai *x* dan y dari dua buah titik. Fungsi heuristik ini dinamakan Manhattan karena di kota Manhattan diAmerika, jarak dari dua lokasi umumnya dihitung dari blok-blok yang harus dilalui saja dan tentunya tidak bisa dilintasi secara diagonal (Adipranata, 2007). Artinya fungsi heuristik ini digunakan untuk kasus dimana pergerakan pada peta hanya lurus (horisontal atau vertikal), tidak diperbolehkan pergerakan diagonal. Persamaan untuk menghitung *manhattan distance* dapat ditulis sebagai berikut :

h(n)= abs(tujuan.x-n.x)+abs(tujuan.y -n.y) (2.1)

Dimana *h(n)* merupakan perkiraan *cost* dari *node* n ke *node* tujuan yang dihitung dengan fungsi heuristik. Variabel *n.x* merupakan koordinat *x* dari *node n*, sedangkan *n.y* merupakan koordinat y dari *node n*. Variabel *tujuan.x* merupakan koordinat *x* dari *node* tujuan dan *tujuan.y* merupakan koordinat *y* dari *node* tujuan. Nilai dari *h(n)* akan selalu bernilai positif.

2. Euclidian Distance

Heuristik ini akan menghitung jarak berdasarkan panjang garis lurus yang dapat ditarik dari dua buah titik. Sehingga fungsi ini dapat diberlakukan untuk simpul-simpul yang terhubung secara vertical, horizontal, ataupun diagonal (Riftadi, 2007). Perhitungannya adalah sebagai berikut :

$$h(n)= \sqrt{(x.n - x.tujuan)^2 + (y.n - y.tujuan)^2} \quad (2.2)$$

Dimana *h(n)* merupakan perkiraan *cost* dari *node* n ke *node* tujuan yang dihitung dengan fungsi heuristik. Variabel *n.x* merupakan koordinat *x* dari *node n*, sedangkan *n.y* merupakan koordinat y dari *node n*. Variabel *x.tujuan* merupakan koordinat *x* dari *node* tujuan dan *y.tujuan* merupakan koordinat *y* dari *node* tujuan. Nilai dari *h(n)* akan selalu bernilai positif.

Terdapat beberapa jenis algoritma pencarian yang tergolong kedalam pencarian heuristik ini, yaitu *Generate and Test, Hill Climbing, Simulated Annealing(SA), Best First Search(BFS), Greedy Best First Search, dan A*.*

2.4. Algoritma A*

Algoritma A* menyelesaikan masalah yang menggunakan graf untuk perluasan ruang statusnya. Dengan menerapkan suatu heuristik, algoritma ini membuang langkah-langkah yang tidak perlu dengan pertimbangan bahwa langkah-langkah yang dibuang sudah pasti merupakan langkah yang tidak akan mencapai solusi yang diinginkan. Algoritma A* membangkitkan simpul yang paling mendekati solusi. Simpul ini kemudian disimpan suksesornya ke dalam list sesuai dengan urutan yang paling mendekati solusi terbaik. Kemudian, simpul pertama pada list diambil, dibangkitkan suksesornya dan kemudian suksesor ini disimpan ke dalam list sesuai dengan urutan yang terbaik untuk solusi.

Algoritma ini akan mengunjungi secara mendalam (mirip DFS) selama simpul tersebut merupakan simpul yang terbaik. Jika simpul yang sedang dikunjungi ternyata tidak mengarah kepada solusi yang diinginkan, maka akan melakukan runut balik ke arah simpul akar untuk mencari simpul anak lainnya yang lebih menjanjikan dari pada simpul yang terakhir dikunjungi. Bila tidak ada juga, maka akan terus mengulang mencari ke arah simpul akar sampai ditemukan simpul yang lebih baik untuk dibangkitkan suksesornya. Strategi ini berkebalikan dengan algoritma DFS yang mencari sampai kedalaman yang terdalam sampai tidak ada lagi suksesor yang bisa dibangkitkan sebelum melakukan runut balik, dan BFS yang tidak akan melakukan pencarian secara mendalam sebelum pencarian secara melebar selesai. A* baru berhenti ketika mendapatkan solusi yang dianggap solusi terbaik.

Algoritma A* menggunakan fungsi heuristic yang menggabungkan jarak estimasi/heuristik [h(n)] dan jarak sesungguhnya/cost [g(n)] dalam membantu penyelesaian persoalan seperti yang terlampir pada persamaan berikut:

$$f(n) = g(n) + \square(n) \qquad (2.3)$$

Dimana :

f(n) : Nilai fungsi heuristik simpul n

$g(n)$: Biaya sebenarnya dari G ke n

h(n) : Biaya perkiraan dari n ke G

Seperti yang dijelaskan sebelumnya, heuristik adalah nilai yang memberi harga pada tiap simpul yang memandu A* mendapatkan solusi yang diinginkan. Dengan heuristik, maka A* pasti akan mendapatkan solusi (jika memang ada solusinya). Dengan kata lain, heuristik adalah fungsi optimasi yang menjadikan algoritma A* lebih baik dari pada algoritma lainnya. Namun heuristik masih merupakan estimasi/perkiraan biasa saja. Sama sekali tidak ada rumus khususnya. Artinya, setiap kasus memiliki fungsi heuristik yang berbeda-beda. (Russell, Norvig, 1995).

Algoritma A* selalu menemukan solusi jika pada suatu masalah memang memiliki solusi. Namun, untuk masalah yang kompleks, misalnya untuk pencarian jarak terpendek pada graph yang terdiri dari 100 juta simpul, A* akan mengalami masalah waktu proses dan waktu yang dibutuhkan. Untuk mengatasi permasalahan itu, maka dirancanglah variasi A* yang sesuai untuk menyelesaikan permasalahan tertentu (Suyanto, 2007). Variasi A* tersebut diantaranya adalah:

2.4.1. Iterative Deepening A* (IDA*)

Sama seperti algoritma A*, algoritma IDA* juga dapat menemukan solusi untuk masalah yang memang memiliki solusi. Namun, karena dilakukan secara iterative, menyebabkan dalam pencarian solusi mungkin membangkitkan simpul-simpul yang sama secara berulang-ulang, sehingga memakan waktu yang relative lebih lama. Disamping itu, IDA* memiliki keunggulan jumlah memori yang dibutuhkan menjadi jauh lebih sedikit. Algoritma ini cocok untuk permasalahan dengan keterbatasan memori, seperti membangun sistem *Personal Digital Assistant.* Tidak sama seperti komputer, perangkat PDA memiliki memori yng lebih kecil dibandingkan dengan computer. Untuk itu, jika kita ingin membuat sistem untuk perangkat ini, maka IDA* akan mampu menjadi solusi yang tepat dalam penyelesaian permasalahannya.

2.4.2. *Simplified Memory Bounded* A* (SMA*)

Untuk menutupi kelemahan algroritma IDA* yang membangkitkan simpul berulang kali sehingga membutuhkan waktu yang lama dalam prosesnya, maka dirancanglah algoritma SMA*. Pada algoritma SMA*, kita dapat membatasi

pencarian hanya sampai pada simpul-simpul yang dapat dicapai dari *root* sepanjang memori masih mencukupi. Misalnya, computer hanya mampu menyimpan 100 simpul, maka kita bisa membatasi pencarian sampai level 99.

2.4.3. *Modified Bi-directional* A* (MBDA*)

MBDA* adalah algoritma A* yang dilakukan dalam dua arah secara serentak. Yakni dari arah simpul awal ke simpul tujuan dan dari arah simpul tujuan ke simpul awal. Proses pencarian akan berhenti jika *Best Node* dari arah simpul asal telah berada dalam senarai *Closed* dari arah simpul tujuan, yang kemudian dilkukan pemeriksaan apakah peru dilakukan pergantian parent terhadap *Best Node* teresbut. Atau pencarian juga berhenti jika sebaliknya, yaitu jika *Best Node* dari arah simpul tujuan telah berada dalam senarai *Closed* dari arah simpul awal.

Dibandingkan algoritma A*, untuk masalah yang besar MBDA* bisa menyelesaikan lebih cepat dan hemat dalam pemakaian memory karena MBDA* membangkitkan jumlah simpul yang lebih sedikit dibandingkan A*.

2.4.4. *Dynamic Weighting* A* (DWA*)

Algoritma ini memberikan sebuah bobot yang dinamis terhadap fungsi heuristic h. Dengan pembobotan dinamis ini, kita mengasumsikan pada awal iterasi, lebih baik pencarian dilakukan kearah manasaja. Dan ketika goal sudah dekat, barulah pencarian dilakukan focus kearah goal.Sehingga algoritma ini adalah algoritma yang optimal tanpa ada batasan waktu dan memori.

2.4.5. *Beam* A* (BA*)

Algoritma ini memberikan sedikit perbedaan tarhadap algoritma A*. Cara kerja algoritma ini adalah dengan membatasi simpul yang bisa disimpan didalam *Open*. Ketika jumlah simpul di open sudah melebihi batas tertentu, maka simpul dengan nilai f terbesar akan dihapus. Namun, kelemahan algoritma ini adalah ketika jumlah simpul maksimum yang bisa disimpan di dalam senarai *Open* lebih sedikit, maka kemungkinan BA* tidak mampu menemukan solusi.

2.5. Modified Bi-Directional A(MBDA)*

MBDA merupakan salah satu variasi pengembangan algoritma A*. Sehingga pada dasarnya algoritma A* dan MBDA adalah sama. Perbedaannya adalah pada A* penelusuran dilakukan satu arah dari state awal ke state goal. Sedangkan MBDA melakukannya dengan dua arah secara serentak, yaitu dari state awal ke akhir dan dari state akhir ke satate awal. Perbedaan lainnya adalah pada algoritma MBDA dilakukan modifikasi pada fungsi heuristik (pada persamaan 2.1). Karena pencarian solusi dilakukan melalui dua arah, maka persamaan fungsi heuristik juga terdiri dari dua jenis, yaitu fungsi heuristik pencarian maju dan fungsi heuristik pencarian mundur (Suyanto, 2007). Sehingga fungsi heuristik untuk simpul n pada pencarian maju (dari simpul *start* ke simpul *goal*) adalah:

$$f = g\ S,n\ +\ \frac{1}{2}[\ n\ -\ ?_g\ n\ ] \qquad (2.4)$$

Sedangkan fungsi heuristik untuk simpul n pada pencarian mundur (dari simpul *goal* ke simpul *start*) adalah:

$$f = g\ G,n\ +\ \frac{1}{2}[?_g\ n\ -\ ?_S\ n\ ] \qquad (2.5)$$

Dimana :

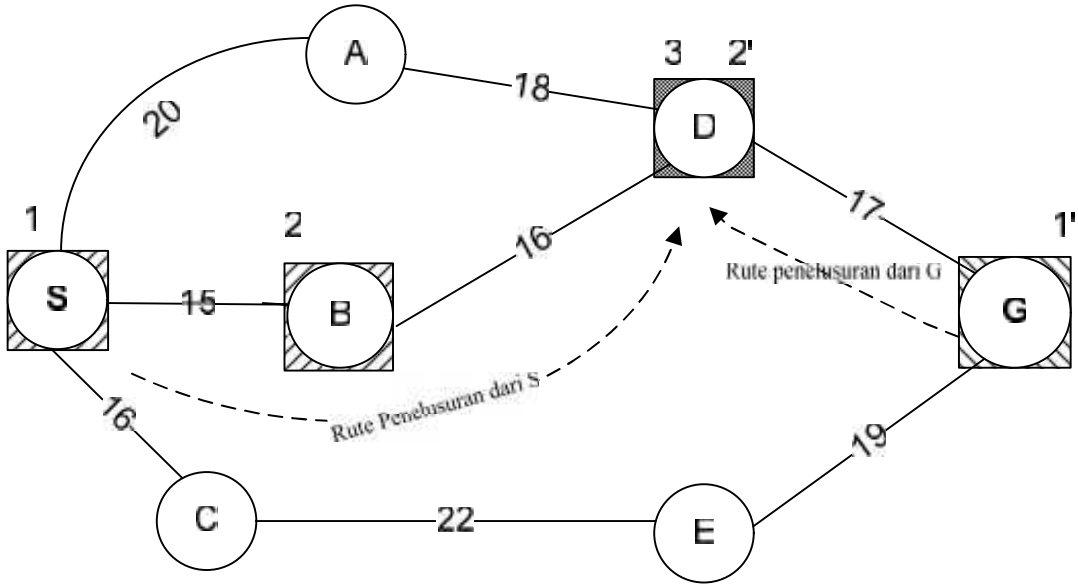
$g(S,n)$: Biaya sebenarnya dari S ke n

$g(G,n)$: Biaya sebenarnya dari G ke n

$h_s(n)$: Biaya perkiraan dari n ke G

$h_g(n)$: Biaya perkiraan dari n ke S

Pencarian pada MBDA akan berhenti di titik tengah dimana pada titik tersebut merupakan pertemuan solusi antara pencarian dari state awal dan pencarian dari state akhir. Berikut adalah gambar yang mensimulasikan bagaimana proses pencarian jalur terpendek menggunakan algoritma MBDA*:



Gambar 2.1. Simulasi pencarian solusi jalur terpendek dengan MBDA*

Dari gambar diatas terllihat bahwa pencarian solusi jalur terpendek dilakukan secara serentak dari dua arah. Yakni dari arah S(*Start*) dan dari arah G(*Goal*). Pada pengulangan pertama, penelusuran dari arah *start* memulai pencarian disimpul S sementara penelusuran dari arah *goal* memulai pencarian disimpul G. Pada pengulangan kedua, penelusuran dari arah *start* memillih simpul B sebagai simpul yang layak sementara penelusuran dari arah *goal* memilih D sebagai simpul yang layak. Pada pengulangan ketiga, penelusuran maju mendapatkan D sebagai simpul yang layak, dan ternyata simpul D merupakan solusi dari pencarian mundur sehingga pencarian berhenti. Hal ini terjadi karena kedua pencarian menemukan titik tengah yakni di simpul D.

Algoritma A* yang memiliki kelemahan dalam menyelesaikan permasalahan tertentu ditutupi oleh algoritma MBDA. Seperti untuk masalah yang lebih kompleks seperti pencarian rute terpendek dari ribuan simpul, performa MBDA akan lebih baik dari pada A*. Berdasarkan penelitian yang dilakukan oleh Tetsuo Shibuya terhadap jaringan jalan di Tokyo membuktikan bahwa jumlah simpul yang dibangkitkan oleh MBDA adalah setengah dari jumlah simpul yang dibangkitkan oleh A*. (Suyanto, 2007).

2.6. Simulasi Graf untuk penjadwalan

Banyak situasi didalam dunia nyata yang dapat direpresentasikan kedalam sebuah abstraksi matematika dimana suatu diagram yang terdiri dari *point-point* dan garis yang menghubungkannya. Hal inilah yang mengawali konsep sebuah struktur data graph (Bondy, Murty, 1979). Secara geometri graf digambarkan dengan sekumpulan titik di dalam bidang dua dimensi yang dihubungkan dengan sekumpulan garis(Shofiyatul Hasanah, 2007). Didalam ilmu komputer, graf merupakan salah satu jenis struktur data yang digunakan dalam pengolahan suatu data. Pada struktur tersebut data akan disimpan dalam bentuk simpul-simpul(vertex) dan setiap data akan dihubungkan dengan suatu sisi(*edge*).

Penjadwalan adalah kegiatan yang dibutuhkan untuk menentukan, mengurutkan, memperkirakan durasi suatu aktifitas dalam suatu kegiatan dengan mempertimbangkan keterbatasan yang ada (Wiwin Oktaviani, 2011). Sehingga dari definisi diatas dapat didefinisikan bahwa pejadwalan penggunaan laboratorium merupakan suatu proses dalam menentukan, mengurutkan, memperkirakan durasi suatu aktifitas penggunaan laboratorium seperti praktikum ataupun perkuliahan yang dilakukan dilaboratorium, dengan mempertimbangkan keterbatasan ruangan dan waktu kosong yang dimiliki mahasiswa ataupun pengajar yang terlibat.

Penerapannya untuk kasus penjadwalan khususnya penggunaan ruang laboratorium, kasus harus direpresentasikan dalam bentuk graf berbobot terlebih dahulu. Untuk merepresentasikan kasus menjadi graf, perlu ditentukan kandidat jadwal yang akan djadikan simpul pada graf. Kandidat jadwal merupakan kemungkinan pasangan waktu, ruangan dan kelas praktikum yang dapat disusun menjadi suatu jadwal.

Contoh kasus pembentukan graf berbobot dan pencarian solusi untuk menemukan hasil akhir penjadwalan dilampirkan melalui tabel 2.1 berikut.

Tabel 2.1. Contoh kasus penjadwalan penggunaan ruang laboratorium

No	Dosen	Waktu kosong dosen	Kelas Praktikum	Waktu Kosong mahasiswa	Waktu Kosong mhs dan dosen (Wm∩Wd)
1	Ali	Senin 08:00-12:10 Selasa 10:30-14:40	X	Senin 08:00-10:30	Senin 08:00-10:30
				Selasa 10:30-12:10	Selasa 10:30-12:10
				Rabu 08:00-12:10	
			Y	Selasa 10:30-12:10	Selasa 10:30-12:10
				Kamis 08:00-10:30	
2	Budi	Senin 08:00-04:10 Selasa 08:00-10:30	Z1	Senin 08:00-10:30	Senin 08:00-10:30
				Jum'at 13:00-16:10	
			Z2	Senin 08:00-12:10	Senin 08:00-12:10
				Selasa 08:00-12:10	Selasa 08:00-10:30
3	Tini	Senin 08:00-10:30 Selasa 08:00-12:10	M	Senin 08:00-12:10	Senin 08:00-10:30
				Selasa 08:00-16:10	Selasa 08:00-12:10
				Rabu 10:30-12:10	

Dari contoh kasus diatas, dinyatakan bahwa mata praktikum X dan Z berjumlah 3 SKS. Sedangkan matapraktikum Y dan M berjumlah 2 SKS. Dinyatakan laboratorium tersebut memiliki 2 ruangan yang aktif yaitu R1 dan R2. Sehingga dari kasus diatas dapat disusun simpul-simpul yang akan menjadi kommpunen graf. Proses penyusunan simpul berdasarkan pengumpulan kemungkinan slot waktu yang dapat menjadi tempat peletakan suatu kelas praktikum. Kemungkinan slot waktu memiliki rentang sesuai dengan nilai irisan antara waktu kosong dosen/asisten dan mahasiswa. Dari rentang waktu tersebut akan dipasangkan dengan kelas praktikum bersangkutan sesuai jumlah SKS masing-masing kelas praktikum.

X → senin $_{1}$ 08:00-10:30

Y → selasa $_{1}$ 10:30-12:10

Z1→ Senin $_{1}$ 8:00-10:30

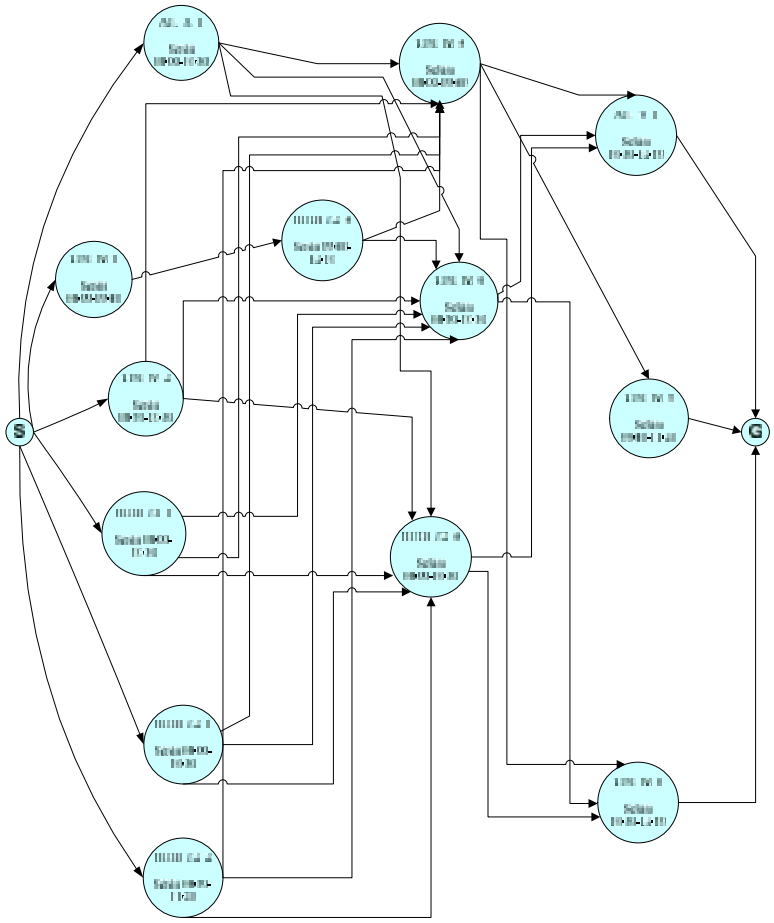
Z2→ Senin $_{1}$ 8:00-10:30, Senin $_{2}$ 08:50-11:20, Senin $_{3}$ 09:40-12:10, Selasa $_{4}$ 08:00-10:30

M → Senin 08:00-09:40 (1), senin 08:50-10:30 (2), selasa 8:00-09:40 (3), Selasa 08:50-10:30 (4), selasa 09:40-11-20 (5), selasa 10:30-12:10 (6)

Simpul yang didapat adalah sebagai berikut:

Simpul = {ALI_X_1, ALI_Y_1, BUDI_Z1_1, BUDI_Z2_1, BUDI_Z2_2, BUDI_Z2_3, BUDI_Z2_4, TINI_M_1, TINI_M_2, TINI_M_3, TINI_M_4, TINI_M_5, TINI_M_6}

Setelah simpul terkumpul, maka langkah selanjutnya adalah menyusun simpul tersebut menjadi suatu kesatuan dalam bentuk graf. Proses penyusunan dilakukan dari kiri ke kanan berdasarkan iterasi waktu hari perhari, yakni dari hari senin sampai hari jum'at dan dari pukul 08:00 WIB sampai 15:30 WIB. Hasil graf yang didapat diilustrasikan pada gambar 2.2.



Gambar 2.2. Graf kasus penjadwalan